          Guidelines for Extending the RTP Control Protocol (RTCP)
                   draft-ott-avt-rtcp-guidelines-01.txt

Status of this Memo

Copyright Notice

Abstract

   The RTP Control Protocol (RTCP) is used along with the Real-time
   Transport Protocol (RTP) to provide a control channel between media
   senders and receivers.  This allows constructing a feedback loop to
   enable application adaptivity and monitoring, among other uses.  The
   basic reporting mechanisms offered by RTCP are generic, yet quite
   powerful and suffice to cover a range of uses.  This document
   provides guidelines on extending RTCP if those basic mechanisms prove

   insufficient.


Table of Contents

1.  Introduction

   The Real-time Transport Protocol (RTP) [RFC3550] is used to carry
   time-dependent (often continuous) media such as audio or video across
   a packet network in an RTP session.  RTP usually runs on top of an
   unreliable transport such as UDP, DTLS, or DCCP, so that RTP packets
   are susceptible to loss, re-ordering, or duplication.  Associated
   with RTP is the RTP Control Protocol (RTCP) which provides a control
   channel for each session: media senders provide information about
   their current sending activities ("feed forward") and media receivers
   report on their reception statistics ("feedback") in terms of
   received packets, losses, and jitter.  Senders and receivers provide
   self-descriptions allowing to disambiguate all entities in an RTP
   session and correlate SSRC identifiers with specific application
   instances.  RTCP is carried over the same transport as RTP and is
   hence inherently best-effort and hence the RTCP reports are designed
   for such an unreliable environment, e.g., by making them "for
   information only".

   The RTCP control channel provides coarse-grained information about
   the session in two respects: 1) the RTCP SR and RR packets contain
   only cumulative information or means over a certain period of time
   and 2) the time period is in the order of seconds and thus neither
   has a high resolution nor does the feedback come back
   instantaneously.  Both these restrictions have their origin in RTP
   being scalable and generic.  Even these basic mechanisms (which are
   still not implemented everywhere despite their simplicity and very
   precise specification, including sample code) offer substantial
   information for designing adaptive applications and for monitoring
   purposes, among others.

   Recently, numerous extensions have been proposed in different
   contexts to RTCP which significantly increase the complexity of the
   protocol and the reported values, mutate it toward an command
   channel, and/or attempt turning it into a reliable messaging
   protocol.  While the reasons for such extensions may be legitimate,
   many of the resulting designs appear ill-advised in the light of the
   RTP architecture.  Moreover, extensions are often badly motivated and
   thus appear unnecessary given what can be achieved with the RTCP
   mechanisms in place today.

   This document is intended to provide some guidelines for designing
   RTCP extensions.  It is particularly intended to avoid an extension
   creep for corner cases which can only harm interoperability and
   future evolution of the protocol at large.  We first outline the
   basic operation of RTCP and constructing feedback loops using the
   basic RTCP mechanisms.  Subsequently, we outline categories of
   extensions proposed (and partly already accepted) for RTCP and

discuss issues and alternative ways of thinking by example.  Finally,
we provide some guidelines and highlight a number of questions to ask
(and answer!) before writing up an RTCP extension.


2.  Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
"SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
document are to be interpreted as described in BCP 14, RFC 2119
[RFC2119] and indicate requirement levels for compliant
implementations.

The terminology defined in RTP [RFC3550], the RTP Profile for Audio
and Video Conferences with Minimal Control [RFC3551], and the
Extended RTP Profile for RTCP-Based Feedback (RTP/AVPF) [RFC4585]
apply.


3.  RTP and RTCP Operation Overview

One of the twelve networking truths states: "In protocol design,
perfection has been reached not when there is nothing left to add,
but when there is nothing left to take away" [RFC1925].  Despite (or
because of) this being an April, 1st, RFC, this specific truth is
very valid and it applies to RTCP as well.

In this section, we will briefly review what is available from the
basic RTP/RTCP specifications.  As specifications, we include those
which are generic, i.e., do not have dependencies on particular media
types.  This includes the RTP base specification [RFC3550] and
profile [RFC3551], the RTCP bandwidth modifiers for session
descriptions [RFC3556], the timely feedback extensions (RFC 4585),
and the extensions to run RTCP over SSM networks.  RTCP XR [RFC3611]
provides extended reporting mechanisms which are partly generic in
nature, partly specific to a certain media stream.

We do not discuss RTP-related documents that are orthogonal to RTCP.
The Secure RTP Profile [RFC3711] can be used to secure RTCP in much
the same way it secures RTP data, but otherwise does not affect the
behaviour of RTCP.  The transport protocol used also has little
impact, since RTCP remains a group communication protocol even when
running over a unicast transport (such as TCP [RFC4571] or DCCP
[I-D.ietf-dccp-rtp]), and is little affected by congestion control
due to its low rate relative to the media.  The description of RTP
topologies [RFC5117] is useful knowledge, but is functionally not
relevant here.  The various RTP error correction mechanisms (e.g.
[RFC2198], [RFC2733], [RFC4588], [RFC5109]) are useful for protecting

RTP media streams, and may be enabled as a result of RTCP feedback, but do not directly affect RTCP behaviour.

3.1.  RTCP Capabilities

The RTP/RTCP specifications quoted above provide feedback mechanisms with the following properties, which can be considered as "building blocks" for adaptive real-time applications for IP networks.

o  Sender Reports (SR) indicate to the receivers the total number of packets and octets have been sent (since the beginning of the session or the last change of the sender's SSRC).  These values allow deducing the mean data rate and mean packet size for both the entire session and, if continuously monitored, for every transmission interval.  They also allow a receiver to distinguish between breaks in reception caused by network problems, and those due to pauses in transmission.
o  Receiver Reports (RR) and SRs indicate reception statistics from each receiver for every sender.  These statistics include:
   *  The packet loss rate since the last SR or RR was sent.
   *  The total number of packets lost since the beginning of the session which may again be broken down to each reporting period.
   *  The highest sequence number received so far -- which allows a sender to roughly estimate how much data is in flight when used together with the SR and RR timestamps (and also allows observing whether the path still works and at which rate packets are delivered to the receiver).
   *  The moving average of the inter-arrival jitter of media packets.  This gives the sender an indirect view of the size of any adaptive playout buffer used at the receiver ([RFC3611] gives precise figures for VoIP sessions).
o  Sender Reports also contain NTP and RTP format timestamps.  These allow receivers to synchronise multiple RTP streams, and (when used in conjunction with Receiver Reports) allow the sender to calculate the current RTT to each receiver.  This value can be monitored over time and thus may be used to infer trends at coarse granularity.  A similar mechanism is provided by [RFC3611] to allow receivers to calculate the RTT to senders.

RTCP sender reports and receiver reports are sent, and the statistics are sampled, at random intervals chosen uniformly in the range 0.5 ... 1.5 times the deterministic calculated interval, T. The interval T is calculated based on the media bit rate, the mean RTCP packet size, whether the sampling node is a sender or a receiver, and the number of participants in the session, and will remain constant while the number of participants in the session remains constant.  The lower bound on the base inter-report interval, T, is five seconds, or
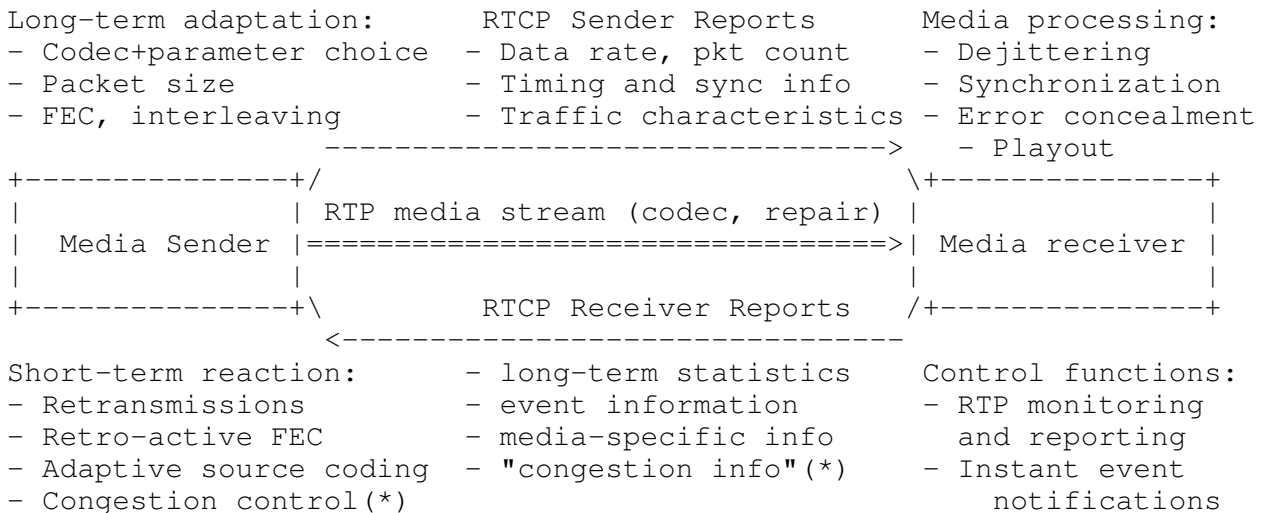
360 seconds divided by the session bandwidth in kilobits/second
(giving an interval smaller than 5 seconds for bandwidths greater
than 72 kb/s) [RFC3550].

This lower limit can be eliminated, allowing more frequent feedback,
when using the early feedback profile for RTCP [RFC4585].  In this
case, the RTCP frequency is only limited by the available bitrate
(usually 5% of the media stream bit rate is allocated for RTCP).  If
this fraction is insufficient, the RTCP bitrate may be increased in
the session description to enable more frequent feedback [RFC3556].
Ongoing work [I-D.ietf-avt-rtcp-non-compound] may reduce the mean
RTCP packet size, further increasing feedback frequency.

The mechanisms defined in [RFC4585] even allow -- statistically -- a
receiver to provide close-to-instant feedback to a sender about
observed events in the media stream (e.g. picture or slice loss).

RTCP is suitable for unicast and multicast communications.  All basic
functions are designed with group communications in mind.  While
traditional (any-source) multicast (ASM) is clearly not available in
the Internet at large, source-specific multicast (SSM) and overlay
multicast are -- and both are commercially relevant.  RTCP extensions
have been defined to operate over SSM, and complex topologies may be
created by interconnecting RTP mixers and translators.  The group
communication nature of RTP and RTCP is also essential for the
operation of Multipoint Conference Units.

These mechanisms can used to implement a quite flexible feedback loop
and enable short-term reaction to observed events as well as long
term adaptation to changes in the networking environment.  Adaptation
mechanisms available on the sender side include (but are not limited
to) choosing different codecs, different parameters for codecs
(spatial or temporal resolution for video, audible quality for audio
and voice), and different packet sizes to adjust the bit rate.
Furthermore, various forward error correction mechanisms and, if RTTs
are short and the application permits extra delays, even reactive
error control such as retransmissions.  Long-term feedback can be
provided in regular RTCP reports at configurable intervals, whereas
(close-to-)instant feedback is available by means of the early
feedback profile.  Figure 1 below outlines this idea graphically.

```
Long-term adaptation:       RTCP Sender Reports     Media processing:
- Codec+parameter choice  - Data rate, pkt count   - Dejittering
- Packet size             - Timing and sync info   - Synchronization
- FEC, interleaving       - Traffic characteristics - Error concealment
                 -------------------------------->   - Playout
+--------------+/                              \+--------------+
|              | RTP media stream (codec, repair) |            |
|  Media Sender |===============================>| Media receiver |
|              |                                |            |
+--------------+\          RTCP Receiver Reports  /+--------------+
                 <-------------------------------
Short-term reaction:      - long-term statistics  Control functions:
- Retransmissions         - event information      - RTP monitoring
- Retro-active FEC        - media-specific info      and reporting
- Adaptive source coding  - "congestion info"(*)   - Instant event
- Congestion control(*)                              notifications
```

   (*) RTCP feedback is insufficient for TCP-Friendly congestion control
       purposes due to the infrequent nature of reporting (which should
       be in the order of once per RTT), but can still be used to adapt
       to the available bandwidth on slower timescales.

                  Figure 1: Outline of an RTCP Feedback Loop

   It is important to note that not all information needs to be
   signalled explicitly -- ever or upon every RTCP packet -- but can be
   derived locally from other pieces of information and from the
   evolution of the information over time.

3.2.  RTCP Limitations

   The design of RTP limits what can meaningfully be done (and hence
   should be done) with RTCP.  In particular, the design favours
   scalability and loose coupling over tightly controlled feedback
   loops.  Some of these limitations are listed below (they need to be
   taken into account when designing extensions):

   o  RTCP is designed to provide occasional feedback which is unlike,
      e.g., TCP ACKs which can be sent in response to every (other)
      packet.  It does not offer per-packet feedback (even when using
      [RFC4585] with increased RTCP bandwidth fraction, the feedback
      guarantees are only statistical in nature).
   o  RTCP is not capable of providing truly instant feedback.
   o  RTCP is inherently unreliable, and does not guarantee any
      consistency between the observed state at multiple members of a
      group.

It is important to note that these features of RTCP are intentional
design choices, and are essential for it to scale to large groups.


4.  Issues with RTCP Extensions

Issues that have come up in the past with extensions to RTP and RTCP
include (but are probably not limited to) the following:

o  Defined only or primarily for unicast two-party sessions.  RTP is
   inherently a group communication protocol, even when operating on
   a unicast connection.  Extensions may become useful in the future
   well outside their originally intended area of application, and
   should consider this.  Stating that something works for unicast
   only is not acceptable, particularly since various flavours of
   multicast have become relevant again, and as middleboxes such as
   repair servers, mixers, and RTCP-supporting MCUs [RFC5117] become
   more widely used.
o  Assuming reliable (instant) state synchronization.  RTCP reports
   are sent irregularly and may be lost.  Hence, there may be a
   significant time lag (several seconds) between intending to send a
   state update to the RTP peer(s) and the packet being received, in
   some cases, the packet may not be received at all.
o  Requiring reliable delivery of RTCP reports.  While reliability
   can be implemented on top of RTCP using acknowledgements, this
   will come at the cost of significant additional delay, which may
   defeat the purpose of providing the feedback in the first place.
   Moreover, for scalability reasons due to the group-based nature of
   RTCP, these ACKs need to be adaptively rate limited or targeted to
   a subgroup or individual entity to avoid implosion as group sizes
   increase.  RTCP is not intended or suitable for use as a reliable
   control channel.
o  Commands are issued, rather than hints given.  RTCP is about
   reporting observations -- in a best-effort manner -- between RTP
   entities.  Causing actions on the remote side requires some form
   of reliability (see above), and adherence cannot be verified.
o  RTCP reporting is expanded to become a network management tool.
   RTCP is sensitive to the size of RTCP reports as the latter
   determines the mean reporting interval given a certain bit rate
   share for RTCP.  The amount of information going into RTCP reports
   should primarily target the peer (and thus include information
   that can be meaningfully reacted upon).  Gathering and reporting
   statistics beyond this is not an RTCP task and should be addressed
   by out-of-band protocols.
o  Serious complexity is created.  Related to the previous item, RTCP
   reports that convey all kinds of data first need to gather and
   calculate/infer this information to begin with (which requires
   very precise specifications).  Given that it already seems to be

      difficult to even implement baseline RTCP, any added complexity
      can only discourage implementers, may lead to buggy
      implementations (in which case the reports do not serve the
      purpose they were intended to), and hinder interoperability.
   o  Architectural issues.  Extensions are written without considering
      the architectural concepts of RTP.  For example, point-to-point
      communication is assumed, yet third party monitors are expected to
      listen in.  Besides being a bad idea to rely on eavesdropping
      entities on the path, this is obviously not possible if SRTP is
      being used with encrypted SRTCP packets.

   This list is surely not exhaustive.  Also, the authors do not claim
   that the suggested extensions (even if using acknowledgements) would
   not serve a legitimate purpose.  We rather want to draw attention to
   the fact that the same results may be achievable in a way which is
   architecturally cleaner and conceptually more RTP/RTCP-compliant.
   The following section contains a first attempt to provide some
   guidelines on what to consider when thinking about extensions to RTP
   and RTCP.


5.  Guidelines

   Designing RTCP extensions requires consideration of a number issues,
   as well as in-depth understanding of the operation of RTP mechanisms.
   While it is expected that there are many aspects not yet covered by
   RTCP reporting and operation, quite a bit of functionality is readily
   available for use.  Other mechanisms should probably never become
   part of the RTP family of specifications, despite the existance of
   their equivalents in other environments.  In the following, we
   provide some guidance to consider when (and before!) developing an
   extension to RTCP.

   We begin with a short check list concerning the applicability of RTCP
   in the first place:

   o  Check what can be done with the existing mechanisms, exploiting
      the information that is already available in RTCP.  Is the need
      for an extension only perceived (e.g., due to lazy implementers,
      or artificial constraints in endpoints), or is the function or
      data really not available (or derivable from existing reports)?
      It is worthwhile remembering that redundant information supplied
      by a protocol runs the risk of being inconsistent at some point,
      and various implementation may handle such situations differently
      (e.g., give precedence to different values).  Similarly, there
      should be exactly one (well-specified) way of performing every
      function and operation of the protocol.

   o  Is the extension applicable to RTP entities running anywhere in
      the Internet, or is it a link- or environment-specific extension?
      In the latter cases, local extensions (e.g. header compression, or
      non-RTP protocols) may be preferable.  RTCP should not be used to
      carry information specific to a particular (access) link.
   o  Is the extension applicable in a group communication environment,
      or is it specific to point-to-point communications?  RTP and RTCP
      are inherently group communication protocols, and extensions must
      scale gracefully with increasing group sizes.

   From a conceptual viewpoint, the designer of every RTCP extension
   should ask -- and answer(!) -- at least the following questions:

   o  How will this new building block complement and work with the
      other components of RTCP?  Are all interactions fully specified?
   o  Will this extension work with all different profiles (e.g. the
      Secure RTP Profile [RFC3711], and the Extended RTP Profile for
      RTCP-based Feedback [RFC4585])?  Are any feature interactions
      expected?
   o  Should this extension be kept in-line with baseline RTP and its
      existing profiles, or does it deviate so much from the base RTP
      operation that an incompatible new profile must be defined?  Use
      and definition of incompatible profiles is strongly discouraged,
      but if they prove necessary, how do nodes using the different
      profiles interact?  What are the failure modes, and how is it
      ensured that the system fails in a safe manner?
   o  How does this extension interoperate with other nodes when the
      extension is not understood by the peer(s)?
   o  How will the extension deal with different networking conditions
      (e.g., how does performance degrade with increases in losses and
      latency, possibly across orders of magnitude)?
   o  How will this extension work with group communication scenarios,
      such as multicast?  Will the extensions degrade gracefully with
      increasing group sizes?  What will be the impact on the RTCP
      report frequency and bitrate allocation?

   For the specific design, the following considerations should be taken
   into account (they're a mixture of common protocol design guidelines,
   and specifics for RTCP):

   o  First of all, if there is (and for RTCP this applies quite often)
      an old-style mechanisms from a different networking environment,
      don't try to directly recreate this mechanism in RTP/RTCP.  The
      Internet environment is extremely heterogeneous, and will often
      have drastically different properties and behaviour to other
      network environments.  Instead, ask what the actual semantics and
      the result required to be perceived by the application or the user
      are.  Then, design a mechanism that achieves this result in a way

      that is compatible with RTP/RTCP.  (And do not forget that every
      mechanism will break when no packets get through -- the Internet
      does not guarantee connectivity or performance.)
   o  Target re-usability of the specification.  That is, think broader
      than a specific use case and try to solve the general problem in
      cases where it makes sense to do so.  Point solutions need a very
      good motivation to be dealt with in the IETF in the first place.
      This essentially suggests developing buildings blocks whenever
      possible, allowing them to be combined in different environments
      than initially considered.  Where possible, avoid mechanisms that
      are specific to particular payload formats, media types, link or
      network types, etc.
   o  For everything (packet format, value, procedure, timer, etc.)
      being defined, make sure that it is defined properly, so that
      independent interoperable implementation can be built.  It is not
      sufficient that you can implement the feature: it has to be
      implementable in several years by someone unfamiliar with the
      working group discussion and industry context.  Remember that
      fields need to be both generated and reacted upon, that mechanisms
      need to be implemented, etc., and that all of this increases the
      complexity of an implementation.  Features which are too complex
      won't get implemented (correctly) in the first place.
   o  Extensions defining new metrics and parameters should reference
      existing standards whenever possible, rather than try to invent
      something new and/or proprietary.
   o  Remember that not every bit or every action must be represented or
      signalled explicitly.  It may be possible to infer the necessary
      pieces of information from other values or their evolution (a very
      prominent example is TCP congestion control).  As a result, it may
      be possible to decouple bits on the wire from local actions and
      reduce the overhead.
   o  Particularly with media streams, reliability can often be "soft".
      Rather than implementing explicit acknowledgements, receipt of a
      hint may also be observed from the altered behaviour (e.g., the
      reception of a requested intra-frame, or changing the reference
      frame for video, changing the codec, etc.).  The semantics of
      messages should be idempotent so that the respective message may
      be sent repeatedly.  Requiring hard reliability does not scale
      with increasing group sizes, and does not degrade gracefully as
      network performance reduces.
   o  Choose the appropriate extension point.  Depending on the type of
      RTCP extension being developed, new data items can be transported
      in several different ways:
      *  A new RTCP SDES item is appropriate for transporting data that
         describes the source, or the user represented by the source,
         rather than the ongoing media transmission.  New SDES items may
         be registered to transport source description information of
         general interest, or the PRIV item may be used for proprietary

extensions.
*   A new RTCP XR block type is appropriate for transporting new
    metrics regarding transmission or reception quality.
*   New RTP profiles may define a profile-specific extension to
    RTCP SR and/or RR packets, to give additional feedback.  Such
    extension is not recommended, since the resulting packets are
    not backwards compatible.  It's generally more appropriate to
    define a new RTCP XR block or a new RTCP packet type instead.
*   New RTCP AVPF transport layer feedback messages should be used
    to transmit general purpose feedback information, to be
    generated and processed by the RTP transport.  Examples include
    (negative) acknowledgements for particular packets, or requests
    to limit the transmission rate.  This information is intended
    to be independent of the codec or application in use.
*   New RTCP AVPF payload-specific feedback messages should be used
    to convey feedback information that is specific to a particular
    media codec, RTP payload format, or category of RTP payload
    formats.  Examples include video picture loss indication or
    reference picture selection, that are useful for many video
    codecs.
*   New RTCP AVPF application layer feedback messages should be
    used to convery higher-level feedback, from one application to
    another, above the level of codecs or transport.
*   A new RTCP APP packet is appropriate for private use by
    applications that don't need to interoperate with others, or
    for experimentation before registering a new RTCP packet type.
    It is not appropriate to register an RTCP APP packet in a
    standards document.
Finally, new RTCP packet types may be registered if none of the
other extension points are appropriate.

The RTP framework was designed following the principle of application
level framing with integrated layer processing, proposed by Clark and
Tennenhouse [ALF].  Effective use of RTP requires that extensions and
implementations be designed and built following the same philosophy.
That philosophy differs markedly from many previous systems in this
space, and making effective use of RTP requires an understanding of
those differences.


6.  Security Considerations

This memo does not specify any new protocol mechansims or procedures,
and so raises no explicit security considerations.  When designing
RTCP extensions, it is important to consider the following points:

   o  Privacy: RTCP extensions, in particular new Source Description
      (SDES) items, can potentially reveal information considered to be
      sensitive by end users.  Extensions should carefully consider the
      uses to which information the release could be put, and should be
      designed to reveal the minimum amount of additional information
      needed for their correct operation.
   o  Congestion control: RTCP transmission timers have been carefully
      designed such that the total amount of traffic generated by RTCP
      is a small fraction of the media data rate.  One consequence of
      this is that the individual RTCP reporting interval scales with
      both the media data rate and the group size.  The RTCP timing
      algorithms have been shown to scale from two-party unicast
      sessions to group with tens of thousands of participants, and to
      gracefully handle flash crowds and sudden departures [TimerRecon].
      Proposals that modify the RTCP timer algorithms must be careful to
      avoid congestion, potentially leading to denial of service, across
      the full range of environments where RTCP is used.
   o  Denial of service: RTCP extensions that change the location where
      feedback is sent must be carefully designed to prevent denial of
      service attacks against third party nodes.  When such extensions
      are signalled, for example in SDP, this typically requires some
      form of authentication of the signalling messages (e.g. see the
      security considerations of [I-D.ietf-avt-rtcpssm]).

   At the time of this writing there are several proposals for in-band
   signalling of secure RTP sessions, where the signalling information
   is conveyed on the media path.  These proposals were discussed in the
   Audio/Video Transport working group session at the 67th IETF meeting,
   with the consensus being that such signalling is not to be conveyed
   within RTP data packets, but should instead be sent within some form
   of control packet, and that it is acceptable to multiplex control and
   data packets on the same port, provided the packet types can be
   clearly distinguished.  There was no consensus in the working group
   on the question of whether keying information should be conveyed in
   RTCP packets multiplexed on the RTP port, or in some other protocol
   multiplexed on the RTP port.  The opinion of the working group chairs
   and area director was, however, that both approaches are workable,
   and fit within the RTP architecture, but that it may be cleaner to
   use a separate keying protocol (modelled after STUN), than to try to
   fit keying within RTCP

   The security considerations of the RTP specification [RFC3550] apply,
   along with any applicable profile (e.g.  [RFC3551]).


7.  IANA Considerations

   No IANA actions are necessary.

8.  Acknowledgements

   This draft has been motivated by many discussions in the AVT WG.  The
   authors would like to acknowledge the active members in the group for
   providing the inspiration.


9.  References

9.1.  Normative References

   [RFC1925]   Callon, R., "The Twelve Networking Truths", RFC 1925,
               April 1996.

   [RFC2119]   Bradner, S., "Key words for use in RFCs to Indicate
               Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC2198]   Perkins, C., Kouvelas, I., Hodson, O., Hardman, V.,
               Handley, M., Bolot, J., Vega-Garcia, A., and S. Fosse-
               Parisis, "RTP Payload for Redundant Audio Data", RFC 2198,
               September 1997.

   [RFC2733]   Rosenberg, J. and H. Schulzrinne, "An RTP Payload Format
               for Generic Forward Error Correction", RFC 2733,
               December 1999.

   [RFC3550]   Schulzrinne, H., Casner, S., Frederick, R., and V.
               Jacobson, "RTP: A Transport Protocol for Real-Time
               Applications", STD 64, RFC 3550, July 2003.

   [RFC3551]   Schulzrinne, H. and S. Casner, "RTP Profile for Audio and
               Video Conferences with Minimal Control", STD 65, RFC 3551,
               July 2003.

   [RFC3556]   Casner, S., "Session Description Protocol (SDP) Bandwidth
               Modifiers for RTP Control Protocol (RTCP) Bandwidth",
               RFC 3556, July 2003.

   [RFC3611]   Friedman, T., Caceres, R., and A. Clark, "RTP Control
               Protocol Extended Reports (RTCP XR)", RFC 3611,
               November 2003.

   [RFC3711]   Baugher, M., McGrew, D., Naslund, M., Carrara, E., and K.
               Norrman, "The Secure Real-time Transport Protocol (SRTP)",
               RFC 3711, March 2004.

   [RFC4571]   Lazzaro, J., "Framing Real-time Transport Protocol (RTP)
               and RTP Control Protocol (RTCP) Packets over Connection-

                 Oriented Transport", RFC 4571, July 2006.

   [RFC4585]  Ott, J., Wenger, S., Sato, N., Burmeister, C., and J. Rey,
              "Extended RTP Profile for Real-time Transport Control
              Protocol (RTCP)-Based Feedback (RTP/AVPF)", RFC 4585,
              July 2006.

   [RFC4588]  Rey, J., Leon, D., Miyazaki, A., Varsa, V., and R.
              Hakenberg, "RTP Retransmission Payload Format", RFC 4588,
              July 2006.

   [RFC5109]  Li, A., "RTP Payload Format for Generic Forward Error
              Correction", RFC 5109, December 2007.

   [RFC5117]  Westerlund, M. and S. Wenger, "RTP Topologies", RFC 5117,
              January 2008.

9.2.  Informative References

   [I-D.ietf-avt-rtcpssm]
              Ott, J., "RTCP Extensions for Single-Source Multicast
              Sessions with Unicast Feedback", draft-ietf-avt-rtcpssm-17
              (work in progress), January 2008.

   [I-D.ietf-avt-rtcp-non-compound]
              Johansson, I. and M. Westerlund, "Support for non-compound
              RTCP, opportunities and consequences",
              draft-ietf-avt-rtcp-non-compound-02 (work in progress),
              February 2008.

   [I-D.ietf-dccp-rtp]
              Perkins, C., "RTP and the Datagram Congestion Control
              Protocol (DCCP)", draft-ietf-dccp-rtp-07 (work in
              progress), June 2007.

   [ALF]      Clark, D. and D. Tennenhouse, "Architectural
              Considerations for a New Generation of Protocols",
              Proceedings of ACM SIGCOMM 1990, September 1990.

   [TimerRecon]
              Schulzrinne, H. and J. Rosenberg, "Timer Reconsideration
              for Enhanced RTP Scalability", Proceedings of IEEE
              Infocom 1998, March 1998.

Authors' Addresses

    Joerg Ott
    Helsinki University of Technology
    Otakaari 5 A
    Espoo, FIN   02150
    Finland

    Email: jo@netlab.hut.fi


    Colin Perkins
    University of Glasgow
    Department of Computing Science
    Lilybank Gardens
    Glasgow   G12 8QQ
    United Kingdom

    Email: csp@csperkins.org